

Model-Based Fault Detection and Diagnosis System for NASA Mars Subsurface Drill Prototype

Edward Balaban
NASA Ames Research Center / QSS
Moffett Field, CA 94035
650-604-5655
balaban@email.arc.nasa.gov

Howard N. Cannon
NASA Ames Research Center
Moffett Field, CA 94035

Sriram Narasimhan
NASA Ames Research Center / USRA
Moffett Field, CA 94035

Lee S. Brownston
NASA Ames Research Center / QSS
Moffett Field, CA 94035

Abstract—The Drilling Automation for Mars Environment (DAME) project, led by NASA Ames Research Center, is aimed at developing a lightweight, low-power drill prototype that can be mounted on a Mars lander and be capable of drilling down several meters below the Mars surface for conducting geology and astrobiology research. The DAME drill system incorporates a large degree of autonomy - from quick diagnosis of system state and fault conditions to taking the appropriate recovery actions - while also striving to achieve as many of the operational objectives as possible.

This paper outlines, on a general level, the overall DAME architecture, equipment, and autonomy package. The main focus, however, is on describing the model-based fault detection and diagnosis system, including the modeling approach, the fault modes handled, and the diagnostic algorithms. The results of the latest field tests, conducted in 2006 in Houghton Crater on Devon Island (a Mars analogue site in Canadian Arctic), are also discussed.¹²

TABLE OF CONTENTS

1. INTRODUCTION.....	1
2. DAME OVERVIEW.....	2
3. MODEL-BASED REASONING AND HYDE.....	3
4. DAME MODEL-BASED DIAGNOSTIC SYSTEM.....	6
5. TESTING.....	8
6. FUTURE WORK.....	10
7. CONCLUDING REMARKS.....	10
ACKNOWLEDGEMENTS.....	10
REFERENCES.....	10
BIOGRAPHIES.....	11
APPENDIX A – DRILL AUGER MODEL IN GME.....	12
APPENDIX B – DRILL BIT MODEL IN GME.....	13

1. INTRODUCTION

Future drilling missions will be the key to answering important questions about Mars, including past climate

history, geology and evolution of the upper crust, distribution and depth of liquid water, and, potentially, to finding evidence of past or present life on Mars. To enable a future Mars drilling mission, NASA has been investigating a variety of promising drilling technologies. The work described in this paper centers around developing a lightweight, low-power drill prototype [1] that can be mounted on a Mars lander and be capable of drilling down several meters below the surface. DAME is developed in collaboration with Honeybee Robotics (creators of the Rock Abrasion Tool on Mars Exploration Rovers), and Georgia Institute of Technology. While the current generation of Mars rovers and landers relies primarily on the vigilance of ground controllers to plan their operations, detect faults, and protect them from mission-endangering situations, the same approach may not always be feasible in a drilling application, where a delay in taking a corrective action may lead to rapid exacerbation of the problem. Thus DAME was designed from the beginning to incorporate a large degree of autonomy, from quick diagnosis of fault conditions to taking the appropriate recovery actions, while also striving to achieve as many of the operational objectives as possible.

Three distinct systems handle the fault detection and diagnosis duties in DAME's autonomy architecture. The first one, developed at NASA Ames Research Center, utilizes the model-based paradigm. The second, also from Ames, uses a more traditional rule-based approach. The third, from Georgia Tech, employs neural networks to perform vibration analysis on the drill in order to detect off-nominal conditions. The three systems work in parallel, in order to provide redundancy and to guard against a wider range of potential problems.

At the core of our model-based system is the Hybrid Diagnostic Engine (HyDE), capable of analyzing both discrete and continuous processes. HyDE uses a model of the drill that describes its components and the modes of operation that these components can assume (both nominal and off-nominal). It also describes what external conditions can cause the components to transition from one mode to another. Throughout the drilling process HyDE tracks the

¹ 1-4244-0525-4/07/\$20.00 ©2007 IEEE.
² IEEEAC paper #1549, Version 2, Updated December 7, 2006

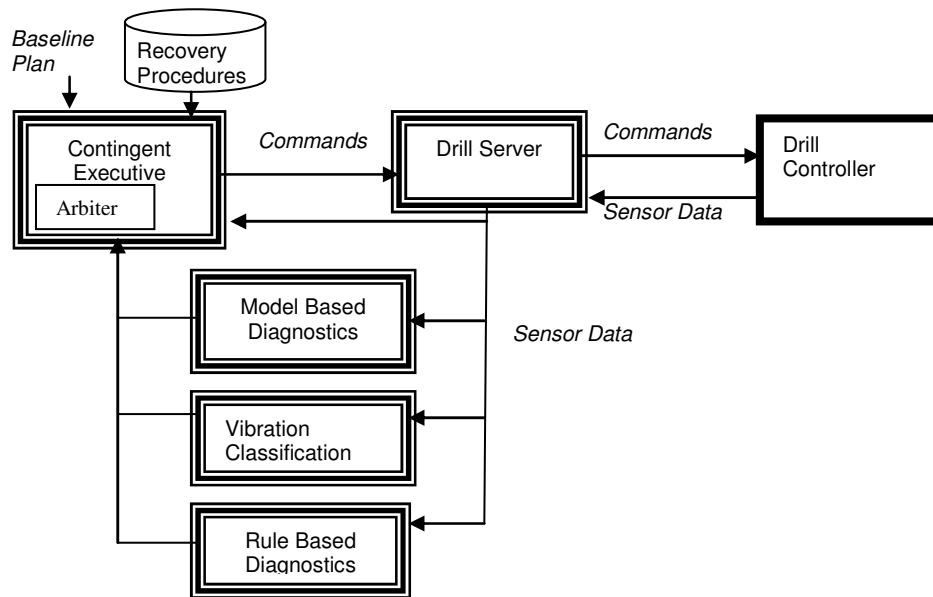


Figure 1: DAME Software Architecture

evolution of the system state, comparing the actual observations to the values predicted by the model for its hypothesized state. If any discrepancies are detected, HyDE searches for suspected faulty components or conditions that can explain the abnormal situation. The results are then passed to the DAME executive system, which decides how to handle the anomaly best.

The paper presents an overview of the DAME project, including its hardware and software architecture and components. The focus then narrows to the DAME model-based diagnostic system, starting with background on HyDE and model-based reasoning. Details on design and development of the model-based diagnostic system follow, describing the control software, the diagnostic model, and the tests conducted to verify and validate them. The testing was conducted in both laboratory and Mars analogue environments. During the latest field tests, conducted in 2006 in Haughton Crater on Devon Island (a Mars analogue site in the Canadian Arctic), the drill operated autonomously for hours at a time and drilled to the depth of over three meters through permafrost, ice, and impact breccia. Finally, we discuss the lessons learned and future research and development plans.

2. DAME OVERVIEW

The Drilling Automation for Mars Exploration (DAME) project was a three-year project started in 2003, funded through the NASA Mars Instrument and Development Program (MIDP) aimed at investigating drilling automation [2]. The goal of this work was to demonstrate “hands off” operation of a Mars-relevant drill system in a Mars analog environment. Because of round trip communication delays to Mars, this technology will be essential to a future Mars drilling mission. The drill system will need to accomplish its tasks and respond to problems as they occur without the

aid of human oversight and intervention.

Each year of the project, the team demonstrated a significant milestone towards its goal. In 2004, the team demonstrated a human-tended, Mars-relevant drill at Haughton Crater on Devon Island in the High Canadian Arctic. Haughton Crater is home to the Haughton Mars Project (HMP), a research station run by the Mars Institute and the SETI Institute dedicated to studying the impact crater and surrounding terrain. The impact crater provides a useful testing ground for the drill because of its similarity to Mars (frozen ice interlaced with frozen soil similar to Mars regolith). This first year of drill testing provided valuable insights into drill failure modes and the recovery actions that the human operators performed. In 2005, the team developed and demonstrated diagnostic software that monitored the human-tended drill system while operating in the crater. Finally, in 2006, the team demonstrated hands-off operation of the drill by adding an onboard executive that responded to diagnosed failure modes by executing recovery procedures. The automated drilling system drilled in the crater for 44 hours to a depth of 3.2 meters with only limited human interaction.

In fact, in a final demonstration of confidence in the system, the drill was left operating unattended for four hours without any human oversight.

DAME Drill System

Figure 2 shows the DAME Mars relevant drilling system developed by Honeybee Robotics [1]. For a drill to be Mars relevant, it must be transportable and thus low in mass (estimate less than 50kg), utilize low power (less than 150 Watts), and cannot use drilling fluids or air to flush the cuttings because of the low atmospheric pressure on Mars. The DAME drill is an auger type drill (for cuttings transport) that used a carbide drag spade bit for the majority of the drilling process. A coring bit was also provided for



Figure 2: DAME Mars Drill System

drilling through hard material or ice. A drill head supported the auger and was raised and lowered along a drill mast (z-axis) via a ball screw driven by a single electric motor. Two electric auger motors attached to the drill head were used for rotating the auger. The primary sensors included optical encoders for sensing position and velocity of the auger rotation and z-axis. Electrical current sensing was used to estimate motor torques and a force transducer was attached to the ball screw to estimate applied force along the z-axis, referred to as Weight on Bit (WOB). The drill also included a temperature sensor in the bit, but the latter was not used for diagnostic purposes.

A key challenge for the diagnostic systems was to infer what was happening in the hole, given the limited amount of parameter sensing available. Likewise, with only limited amounts of power and hence torque available, the drill could easily get stuck or get into situations where limited or no progress could be made. For instance, one of the key problems encountered was referred to as auger choking (Figure 3). This is a condition where the cuttings can fill up inside the auger, causing the drilling process either to be slow, or in extreme conditions, could result in getting stuck.

Architectural Overview

Figure 1 shows the software architecture for the DAME system. The Drill Controller (far right) is the low-level control system responsible for controlling the drill motors and retrieving and converting the sensor signals into engineering units. This sensor data is supplied to the drill server, which either broadcasts the information to the other modules, or provides it upon request. The three diagnostic modules (Model Based, Vibration Classification, and Rule Based) use this data to estimate the state of the drill system. The state is represented as a set of fault modes with associated probabilities. These estimates are provided to the Contingent Executive. An Arbiter function within the Contingent Executive combines the fault probabilities to determine whether to recommend a recovery procedure. The Contingent Executive normally executes a Baseline

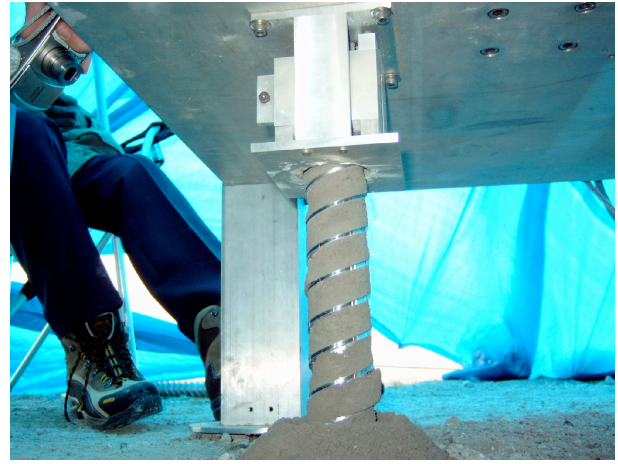


Figure 3: Example of Auger Choking

plan, which may consist of a number of “drill to depth” operations interspersed with science measurements. If a recovery procedure is recommended, it pauses the baseline plan, and inserts the recovery procedure. Once the recovery procedure is completed, the Contingent Executive resumes the baseline plan. The Contingent Executive executes the baseline plan and recovery procedures by sending commands and drilling parameter modifications through the Drill Server to the Drill Controller.

DAME included three diagnostic modules in order to study the relative advantages and disadvantages of various approaches. A Vibration Classification Module, developed by the Georgia Institute of Technology, employed a laser vibrometer to monitor and characterize the drill string vibration signatures. It used a neural network that was trained on data observed and modeled for various fault modes. Another diagnostic approach, referred to as the Rule Based Diagnostic Module, simply compared the magnitude and duration of certain sensor signals to a set of pre-defined thresholds for detecting faults. The magnitude and time at which the thresholds were exceeded determined the probability of the reported fault. The third approach, which is the subject of this paper, is the Model Based Diagnostic System. The model-based approach compares the sensor data to predicted values from a model of a nominal drilling process. When deviations occur, a number of fault modes are searched to determine the best and most likely fit.

3. MODEL-BASED REASONING AND HYDE

Model-based reasoning is an area of Artificial Intelligence that infers information about the performance of a physical system by comparing measurements of it with predictions from a theoretical model of the system. The model-based reasoning paradigm has been successfully applied in the fields of system state diagnosis, planning and scheduling, execution, and prognosis.

HyDE [4][5][6], the model-based diagnostic engine used for

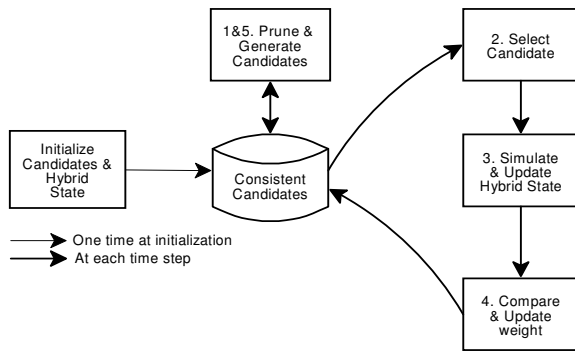


Figure 4: Candidate Management

DAME, was developed by the Discovery and System Health group at NASA Ames Research Center, where research into model-based methods has been conducted for many years. HyDE is the successor to the Livingstone and Livingstone 2 diagnostic engines [3], also developed at NASA Ames and its name stands for “Hybrid Diagnostic Engine”. The Livingstone family of diagnostic engines relied on discrete modeling methods best suitable to such domains as digital electrical systems or computer networks. HyDE supports modeling of both discrete and continuous processes, thus the designation ‘hybrid’. It is well suited to the drilling domain, where discrete control systems are coupled with complex mechanical systems exhibiting continuous behavior.

At the highest level HyDE’s reasoning process involves the management of a set of consistent *candidates*. A candidate is a trajectory that the system is hypothesized to have followed. The trajectory consists of the hypothesized *hybrid state* of the system at all time steps that HyDE has reasoned about. The hybrid state includes the modes of all components at all such time steps (and implicitly any transitions from one mode to another that may have occurred) and the values of all variables represented in the model. A candidate is considered *consistent* if the hybrid state at all time steps consists of values for observable variables that are consistent with their observed values at the corresponding time step.

Figure 4 illustrates the candidate management strategy used by HyDE. When HyDE is started, it has to be initialized with an initial candidate set. For each of the initial candidates, an initial hybrid state also needs to be specified. In the current implementation, HyDE automatically creates a candidate set with only the *nominal* candidate. Nominal candidates are candidates whose trajectories do not include any unobserved (that is, fault) transitions from a nominal mode to a non-nominal mode. HyDE can be easily extended to allow for an initial candidate set that contains multiple candidates some of which may already contain unobserved transitions.

Once the candidate set has been initialized, the same sequence of operations is performed on this candidate set at each time step that HyDE reasons about. HyDE reasons about a time step upon request, typically when observations have been reported for that time step.. The set of

observations includes values for *input variables* and values for *output variables*. Input variables are properties of the system that drive the behavior evolution. These include commands from the controller to change the configuration of the system as well as external inputs to the system (such as sources) that are sensed. Output variables represent all other sensed properties of the system. Additionally HyDE can be forced to reason about specific time steps that there is no telemetry for.

Prune & Generate Candidates

The first operation consists of pruning candidates that do not satisfy consistency criteria and generating new candidates to fill the candidate set with candidates up to a user-specified maximum. The number of candidate generated can be limited by candidate count, candidate size (the number of fault transitions), candidate probability, or reasoning time.

HyDE generates new candidates only if at least one candidate from the candidate set was found to be inconsistent at some time step. Candidate generation tries to “extend” inconsistent candidates (in an effort to make them consistent) by augmenting their trajectory with *unguarded* transitions, that is, transitions from nominal modes to non-nominal modes.. The search for unguarded transitions is based on what caused the inconsistency in the first place and is described in more detail later in the paper.

The prune-and-generate procedure may have to be repeated when new candidates are added to the consistent candidate set to handle the situation where the added candidates are themselves inconsistent.

Simulate system behavior

To check for the consistency of a candidate, the behavior of the system is simulated using the *behavioral model*, that is, the part of the model that specifies how the components behave in each mode.. First HyDE selects the candidates to check for consistency at the current time step. Typically all candidates are checked for consistency. It is possible, however, to limit the number of candidates tested at each

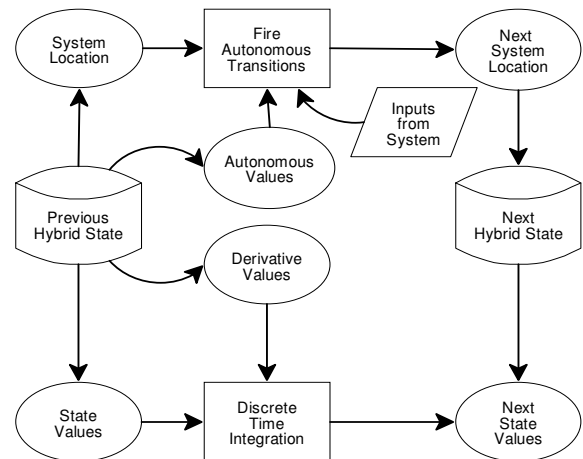


Figure 5: Simulation across time steps

processing step or provide a maximum computational time allowed at each processing step. HyDE stops its reasoning at the current processing step when this limit is reached. It does, however, continue from where it left off in future processing steps.

For the candidates selected for testing, simulation to check for consistency is performed in two steps. First, the hybrid state at the beginning of the current time step is determined based on the hybrid state at the end of the previous time step. This is illustrated in Figure 5. To update the system location (the set of modes of all components), any possible autonomous transitions from the current system location are first evaluated, using the stored values of variables appearing in condition predicates on transitions that are autonomous (that is, not commanded). Any transition whose transition condition is true is fired. Firing a transition implies setting the mode of the component to be the destination mode of that transition. If there is uncertainty about whether a transition is enabled, then a probability of that transition is computed and the transition may be fired if the probability is above a certain threshold. If input variables appear in autonomous transitions, their current values are used instead of the stored values used for other autonomous variables.

The state variables also need to be integrated to determine their values at the current time step. This requires the values of derivative variables, which are also stored in the hybrid state. HyDE uses the Euler method for integration.

The second step of simulation is at the current time step. This is illustrated in Figure 6. First the currently-active model has to be determined. This is done by teasing out the modes of the individual components and then gathering the behavior model fragment associated with that mode. The model fragments are then combined to create the simulation model for the current system location. For example, if the behavior model is represented by constraints, the set of constraints can be solved to obtain the state space equation which may be used as the simulation model. Alternately the constraints may be solved directly using constraint propagation as the simulation mechanism. Using this approach, it is not necessary to pre-compile the simulation

model for each system location. Rather, the model is compiled lazily when necessary. HyDE caches the compiled models so that, when a candidate predicts the same system location for the system, the model does not have to be compiled again.

The simulation may also be stochastic in nature. For example if all the values are represented as distributions then the simulation will result in values that are also distributions rather than single values. This feature can be exercised in HyDE by selecting the Kalman Filter for use as the simulator.

Compare against Observations

The simulation step attempts to compute values for all the variables in the system. The actual values of a subset of these variables are observed through sensors in the system. The consistency of a candidate is checked by comparing these observed values with the predicted values for the same variables from the simulation. A variety of comparison schemes can be implemented depending on the type of values (*e.g.*, boolean, real etc.), the representation of values (single value, distribution etc.), the noise in the sensors, and so on. HyDE currently implements a simple scheme that checks if the predicted and observed value are within a certain range of each other (exact matches are required for boolean and enumeration variables). If necessary, the user may choose to implement a custom comparison algorithm to represent more sophisticated strategies.

The comparison operation is responsible for determining the degree of fit between observed and predicted values. The comparison algorithm used in HyDE is only able to make a binary decision on the degree of fit. If the observation and prediction match then 1 is returned and if they do not match a 0 is returned. Observed variables are marked as consistent or inconsistent depending on this binary decision. If user-supplied algorithms are used for comparison then it is possible to generate graded (0-1) values to better indicate the degree of fit. This information may be used to guide the candidate search.

Candidate Generation

In order to generate new candidates HyDE tries to extend existing candidates that became inconsistent by adding unguarded transitions to them. Whenever a candidate becomes inconsistent, it creates a candidate generator associated with it. Thus the number of candidate generators at any point will be equal to the number of candidates that have become inconsistent with the observations. The candidate generator will be responsible for creating one new best candidate based on the criteria used for determining the best candidates (the most probable, for example). The best candidate among all the ones provided by all of the candidate generators will be selected for consideration next. If it is consistent with the observations, it is retained in the consistent candidates set. If it is not, then a candidate generator associated with the candidate is created and the candidate itself is discarded. The next best candidate is selected from the pool of candidate generators for

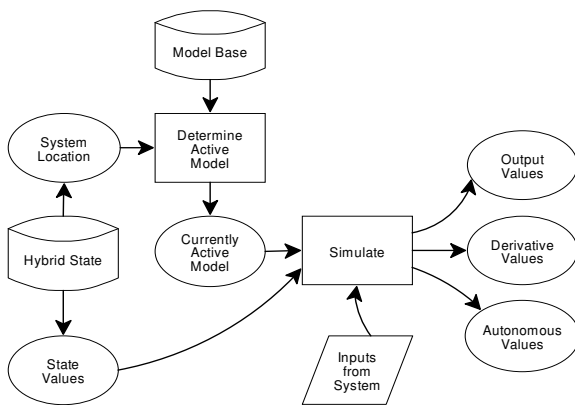


Figure 6: Simulation during a time step

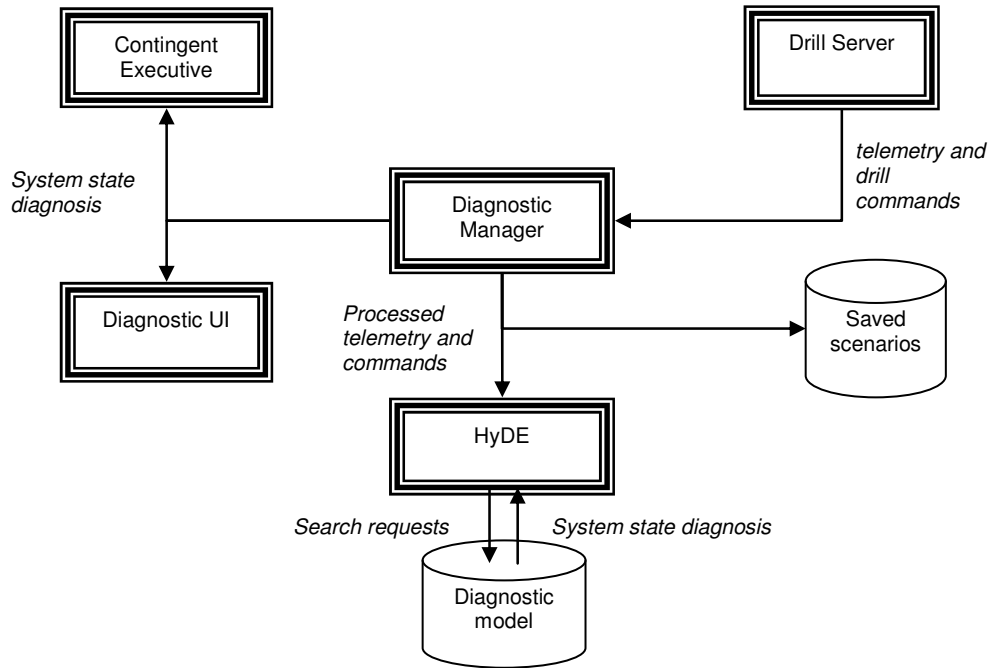


Figure 7: DAME model-based diagnostic architecture

consideration. This process continues until the requisite number of consistent candidates is reached.

To provide the next best candidate, a candidate generator uses conflict resolution. A *conflict* is a set of transitions that, if taken by the simulation of the model, would result in an inconsistency with the observations. When an observation is found to be inconsistent with a candidate's predictions, it is possible to determine the set of transitions that contributed to the mismatch. Given that the detection of the inconsistency may not happen at the same time step as the occurrence of the fault (due to sensor noise, modeling approximations, *etc.*), the conflict should contain transitions from the current time step and also from a few previous time steps. The number of previous time steps that HyDE should consider can be controlled by setting the history parameter as well as other parameters that restrict looking at time steps prior to the last controller command.

If more than one observation is inconsistent, HyDE can create a conflict for each inconsistent observation. In trying to generate a new candidate, HyDE then selects transitions to add to the candidate such that, for each conflict, there is at least one transition that resolves the conflict. A transition resolves a conflict if it is a sibling of some other transition in the conflict. Sibling transitions have the same source mode, but different destination modes. HyDE uses this approach to set up a candidate generator as a set of conflicts and the next best candidate is generated as a set of transitions that resolve all conflicts. HyDE currently uses a best-first search, where best is defined as having the most probable set of transitions.

Although currently not implemented, it is possible to use other search strategies, such as A*.

4. DAME MODEL-BASED DIAGNOSTIC SYSTEM

Architecture

An overview of the diagnostic system architecture and data interchange is given in Figure 7. The two main components of the system are the diagnostic engine (HyDE), described in the previous section, and Diagnostic Manager - the data processing and control module.

The Diagnostic Manager is responsible for the following functions:

- **Obtaining raw telemetry** (observations and commands) from the Drill Server
- **Processing raw telemetry** as appropriate; this includes separating values being used in the model from the general set, filtering to reduce noise and converting them to the format suitable for HyDE
- **Commanding HyDE** to initialize, load the model and the initial conditions, enter new observations or commands, establish a new time step, estimate system state, search for fault candidates if an abnormal condition is detected, return the values of desired internal variables, reset, *etc.*
- **Reporting fault candidates** (if any) to the Executive for performing recovery actions and the Diagnostic UI for display to the operator

- **Storing execution scenarios** for later playback and analysis. All of the sensor data and commands, along with HyDE-specific commands and timestamps are automatically stored in a scenario file that can be used to confirm correctness of a diagnosis, plot the evolution of the system state, *etc.*

To maximize performance, the above operations are parallelized to the extent possible. The fault candidate search time can be limited directly by specifying a timeout or indirectly by specifying the maximum number of fault candidates returned, the maximum number of suspected components per candidate or the minimum probability of a fault candidate considered.

If a fault condition is detected and fault candidate(s) found, they are reported in the following format:

```
<candidate number>
<timestamp>
<probability>
<component 1> <state>
...
<component N> <state>
```

The Diagnostic Manager also performs high-level monitoring of the returned diagnostic results and can reset HyDE automatically if it reports difficulty converging on meaningful results given the current model and data set. This allows the overall operation to continue without interruption and the model to be analyzed and modified after the operation is over.

Telemetry Processing:

The telemetry set coming back from the drill consisted of a wide range of parameters (thirty five total), from drive motor currents to the temperature at the tip of the bit. The diagnostic system utilized a smaller subset of them:

- Auger angular velocity
- Auger torque
- Z-direction force
- Z-direction velocity
- Current drill depth
- Maximum desired rate of penetration
- Drill operating mode (drilling, idle, *etc.*)

A median filter was applied to all the real-valued parameters to reduce the effect of normal fluctuations in sensor readings. Telemetry frames were transmitted at the frequency of 4 frames a second. While this rate did not present any performance problems when the diagnostic system was running on an average laptop computer (Pentium 4, 1.6MHz, 512MB RAM), it was still made adjustable by the operator, in case the system needed to run in a limited computational resources environment.

Diagnostic Model

First generation – 2005. The first generation of the drill model was based on automatic classification of the hardness of the material being drilled through (soft, medium, or hard). After the material hardness was estimated, bit torque prediction was computed as a function of material hardness from the laboratory test results. Auger torque was then estimated as the difference between total torque applied by the drill motors and the torque sensed at the bit. Depending on the material hardness prediction, comparison of the predicted auger torque to the sensed value allowed to determine whether the auger of the drill was in nominal conditions, choking, or binding.

After testing the system with the above model on Devon Island, the following conclusions were reached:

- Variations in auger torque magnitude throughout the drilling process made estimation of the material

Nominal State	Description
Auger Idle	The auger is stationary or spinning idly in or out of the drill hole. There is no movement along the z axis. The drill controller indicates it is in idle mode as well.
Auger Nominal	The drill is either drilling or moving in or out of the hole. The rate of penetration, the torque measured, and other parameters are within nominal expectations for the power applied.
Auger Overload	The auger flights start filling up with cuttings; the auger torque measured is increasing, but is still within nominal limits for the current power output. The rate of penetration remains close to the desired.
Bit Idle	No torque is indicated at the bit and the drill controller is signaling idle mode as well.
Bit Nominal	The torque measured at the bit is within the expected range for the current power output. The rate of penetration remains close to the desired.

Table 1: DAME model nominal modes

hardness in the model overly complicated and unreliable.

- Auger torque predictions could be made through analysis of other parameters, without having to estimate material hardness.
- Magnitudes of auger torque estimated and measured were not a good indicator of how to recover from a particular fault condition.
- **Second generation – 2006.** Given the lessons learned during the 2005 field season, the diagnostic model was redesigned from the ground up, to focus more on describing symptoms of the various fault conditions the drill may encounter in greater detail, rather than trying to classify the relative hardness of the current material layer and simulating the system response from that. Determination whether relatively hard material was encountered was retained, but now more in a sense of a fault mode, to describe the condition when the drill with the current drill bit is not making significant penetration progress for a given period of time. When detected, a bit change from regular to a coring one could be performed as a recovery action. Additional fault modes were added, such as “auger corkscrewing” and “bit inclusion”. The nominal modes modeled are described in more detail in Table 1, the fault modes in Table 2.

5. TESTING

Four major rounds of tests for the integrated drill/autonomy system were conducted in 2005-2006. Their details are provided below:

2005 Laboratory Tests at Honeybee Robotics:

Goals: Integrate components of the autonomy system. Collect data to identify fault signatures and refine the initial take on the model. Test identification of material hardness.

Methodology: The diagnostic system was connected to the drill telemetry server and monitored drill performance while it operated. Detailed telemetry records were kept for subsequent identification of mode signatures. No automated recovery actions were attempted.

Setup: Limestone was utilized as a soft material example, gossan as a medium material, and Devon Island simulation material (composed as a frozen mix of top layer soil from the island, water ice, and impact breccia) in the role of hard material. To keep the latter from melting, a liquid nitrogen cooling system was used. A variety of rpm and WOB regimes were tested for each of the materials. See Figure 8 for an example; the different colors indicate the different levels of weight on bit.

Results: Both choking and binding faults were encountered. The first version of the model was able to characterize the material successfully roughly 40% of the time and detect the faults correctly in about 50% of the cases. The false positive rate was roughly 25%. Enough data was collected during the tests for further refinement of the model.

2005 Field Test in Houghton Crater, Devon Island:

Goals: Test the drill, the model-based, and the neural-net - based diagnostic systems in the Mars analogue environment. Calibrate and improve the diagnostic systems throughout

Fault State	Description	Recovery Procedure
Auger Binding	The auger is rubbing up against something along its length which results in an increased auger torque. The torque on the bit is decreasing.	Raise drill while rotating at high speed.
Auger Choking	Cuttings are accumulating near the bit and are not flowing up the auger flutes. The cuttings expand and cause an increase in auger torque. The torque at the bit is increasing as well.	Slowly raise drill and reverse rotation to clear cuttings.
Bit Jamming	The auger can no longer rotate due to the bit jamming against a rock. The torque at the auger is nominal or below nominal.	Decrease set force. Raise and reverse till auger free.
Bit Inclusion	A pebble, or rock, at the bottom of the hole causes periodic torque spikes, roughly at the frequency of the auger rotation.	Raise drill and then slowly lower to shave a flat surface or pick up rock.
Bit Hard Material	Minimal rate of penetration, even though the auger torque is low and weight on bit is high.	Increase set force. If at max set force, change to coring bit.
Auger Corkscrewing	Auger flutes catch on protruding material and begin to screw into the ground. Identified by large tensile force on drill strings.	Stop, reverse rotate and raise at auger pitch till free. Then up down motion to shear off protrusion.

Table 2: DAME models fault modes

the deployment. Collect data on fault conditions encountered for further modeling work.

Methodology: The two diagnostic systems ran in advisory mode only. Recovery from fault conditions was to be done by the drill operator. Log files for each of the systems and drill telemetry were preserved for subsequent verification of their diagnoses.

Setup: Drilling was done through layers of materials of varying hardness (permafrost, ice, breccia, *etc.*). The drill operated under constant operator supervision. A variety of rpm and WOB regimes were employed to maximize the rate of penetration.

Results: The diagnostic results of this round of tests were mixed. While in many cases the fault conditions were correctly recognized and the relative hardness of the material classified appropriately, consistent identification and diagnosis proved to be difficult for the reasons outlined in the previous section. A decision was made to change the modeling approach from one relying on material classification to one based on deeper analysis of individual fault modes. By the end of the testing, the depth of 2.06 meters was reached and substantial amount of data on fault conditions and the necessary recovery actions was collected.

2006 Laboratory Tests at NASA Ames Research Center:

Goals: Test the modifications to the drill hardware,

especially the newly-added bit torque and temperature sensors. Test the newly-redesigned model for the model-based diagnostic system, the new neural net based system and the rule-based diagnostic system. Integrate the above with the latest communication software and the Contingent Executive. Evaluate the performance of individual systems, as well as the overall autonomy package in preparation for the field deployment on Devon Island.

Methodology: The three diagnostic systems were first tested and calibrated individually, to assess their particular strengths in recognizing specific fault conditions.

Setup: Limestone and Devon Island simulated material (similar in composition to the one in 2005 laboratory testing) were used. The drill system was allowed to operate autonomously for periods of up to two hours.

Results: The new modeling approach proved quite promising and provided a 70-75% rate of fault detection and identification for the faults encountered. The addition of the drill bit torque sensor proved valuable in increasing the diagnostic accuracy. Several areas of the model needing further work were identified and subsequently redesigned.

2006 Field Test in Houghton Crater, Devon Island:

Goals: Test the drill and the integrated autonomy package in the Mars analogue environment and demonstrate “hands

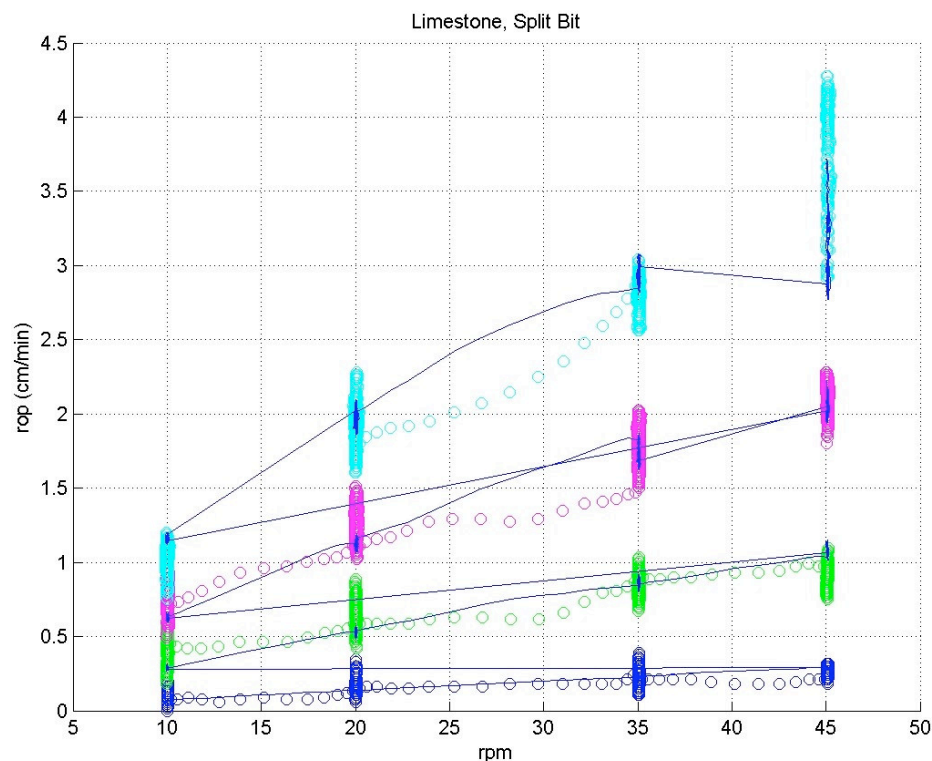


Figure 8: An example of test data collected during laboratory tests for model development

off” operations for extended periods of time (on the order of three hours or more).

Methodology: The three diagnostic systems were independently monitoring the behavior of the drill and reporting their results to the Executive. The latter analyzed and compared the inputs, deciding on the course of the corrective action if a fault was detected. Log files for each of the systems and drill telemetry were preserved for subsequent verification of their actions.

Setup: The drill was positioned on the rim of the Haughton impact crater. Drilling was done first through the top layer of permafrost, then through layers of ice and breccia. The drill strings were added manually, when needed. Drill bits were also interchanged manually, otherwise the drill and its software system (after the initial calibration and modifications) were allowed to operate autonomously.

Results: All of the modeled fault modes were encountered in the field; some, such as choking, binding, and hard material, numerous times. The model-based diagnostic system was able to successfully identify the faults in roughly 85% of the cases. The rate of false positive diagnoses was approximately 5%. The combination of the three different diagnostic systems and the arbitrating mechanism implemented in the Executive resulted in nearly a 100% of the fault conditions being detected and a low-single-digits overall false-positive rate. The drill reached the depth of over 3.2 meters by the end of the test period and operated fully autonomously for periods of up to four hours.

6. FUTURE WORK

While the DAME project has come to its conclusion, the products and methodology developed during it will be utilized on other upcoming planetary drilling and spacecraft autonomy projects.

The lessons learned from the Contingent Executive are being incorporated into the new Universal Executive and its command language, PLEXIL [7], being developed jointly by NASA Ames, NASA Jet Propulsion Laboratory and Carnegie Mellon University.

Work is currently being started to generalize the Diagnostic Manager for a wider range of applications, improve its performance and increase its Technology Readiness Level (TRL). It is also being included into the next generation of integrated spacecraft autonomy architecture, currently under development at NASA Ames.

For HyDE, future efforts will focus on including support for diagnosis of parametric faults. This will involve isolation of potentially deviating parameters and estimating the new values for these parameters. Another planned direction of

work is to develop a methodology for using diagnostic models to automatically determine the recovery actions, rather than having to encode them manually for each fault. An effort is also in process to include alternate modeling and simulation strategies (for example bond graphs) in the HyDE architecture, so as to supply users with a larger selection of algorithms from which to choose.

7. CONCLUDING REMARKS

The DAME project successfully demonstrated the feasibility of deploying a lightweight, low-power subterranean drill on a planetary lander by testing it in relevant laboratory and analogue environments. During the tests, DAME autonomy software monitored system performance through its entire period of operations and was able not only to quickly detect and identify a wide range of fault conditions, but also effectively commanded the drill in recovery from them. This effort also demonstrated that HyDE is becoming a mature general-purpose diagnostic engine, suitable for adoption in a variety of applications.

ACKNOWLEDGEMENTS

The authors would like to acknowledge the contributions of their DAME colleagues who devoted their considerable talents and energy to making this project a success (in alphabetical order):

NASA Ames Research Center: Mark Branson, Scott Christa, Samantha Domville, Brian Glass, David Hall, Sarah Huffman, and Pascal Lee. **Honeybee Robotics:** Kiel Davis, Gail Paulsen, and Kris Zakny. **Georgia Institute of Technology:** Sathya Hanagud, Vin Sharma, Shannon Statham, Alessandro Spadoni, and Agnivesh Tomar

Special thanks also go out to the staff of the Haughton Mars Project camp on Devon Island and the Inuit communities of Resolute Bay and Griese Fiord.

REFERENCES

- [1] G. Paulsen, K. Zacny, P. Chu, E. Mumm, K. Davis, S. Frader-Thompson, H. Cannon, B. Glass, "Robotic Drill Systems for Planetary Exploration", Proc. AIAA Space 2006, San Jose, CA, September 2006
- [2] B. Glass, H. Cannon, M. Branson, S. Hanagud., and G. Paulsen, "DAME: Planetary-Prototype Drilling Automation", Proc. AIAA Space 2006, September 2006, San Jose, CA,
- [3] J. Kurien and P. Nayak, "Back to the Future for Consistency-based Trajectory Tracking," *Proceedings of*

7th National Conference on Artificial Intelligence, 2000.

[4] E. Benazera and S. Narasimhan. "An Extension to the Kalman filter for an Improved Detection of Unknown Behavior", In Proc. of the American Control Conference (ACC-05), Portland, Oregon, June 2005.

[5] S. Narasimhan, R. Dearden, and E. Benazera. "Combining Particle Filters and Consistency Based Approaches for Monitoring and Diagnosis of Stochastic Hybrid Systems", In Proc. of the 15th Intl. Workshop on Principles of Diagnosis (DX 04), pages 123-128, Carcassonne, France, June 23-25, 2004.

[6] S. Narasimhan, L. Brownston, and D. Burrows. "Explanation Constraint Programming for Model-based Diagnosis of Engineered Systems", in Proc. of IEEE Aerospace Conference, Big Sky, Pages 3495-3501, Vol. 5, Montana, March 2004.

[7] V. Verma, A. Jónsson, R. Simmons, T. Estlin, R. Levinson, "Survey of Command Execution Systems for NASA Robots and Spacecraft", "Plan Execution: A Reality Check" Workshop at The International Conference on Automated Planning & Scheduling (ICAPS), 2005

BIOGRAPHIES



Edward Balaban is a researcher in the Diagnosis and System Health group at NASA Ames Research Center. His main area of interest is model-based diagnosis of physical systems. During his years at Ames he has participated in research and development of diagnostic and other autonomy elements for the X-34 experimental reusable launch

vehicle, International Space Station, robotic astronaut assistants, autonomous planetary drills, and the future generation of autonomous micro-spacecraft. He received a Bachelor degree in Computer Science from The George Washington University and a Master degree in Electrical Engineering from Cornell University.



Howard Cannon is a Computer Engineer in the Autonomous Systems and Robotics Area, Intelligent Systems Division at NASA Ames Research Center. He has led and participated in projects related to model based diagnosis, condition based execution for planetary rovers, autonomous drilling operations, and

autonomous mining operations. Prior to joining NASA, he worked at Caterpillar, Inc., developing sensors, actuators, and robotic equipment. He holds an MS in Robotics from

the School of Computer Science at Carnegie Mellon University (1999), and a BS in Mechanical Engineering from Bradley University in Peoria, Illinois (1988).



Dr. Sriram Narasimhan is a computer scientist with University of California, Santa Cruz, and works as a contractor at the NASA Ames Research Center where he is a member of the Modeling Learning and Control group in the Computations Sciences division. He earned his Ph.D. from the Computer Science department at

Vanderbilt University in August 2002. His doctoral dissertation was on model-based diagnosis of hybrid systems. Prior to that, he received his B.E in Computer Science from Birla Institute of Technology & Science in 1995, his M.Sc. in Economics from Birla Institute of Technology & Science in 1995, and his M.S in Computer Science from Vanderbilt University in 1998. Dr. Narasimhan's main research interest is in model-based diagnosis, with a focus on hybrid systems (combination of discrete and continuous behavior). At the NASA Ames Research Center, he is the principal investigator for the research effort to develop HyDE a model-based diagnosis engine for hybrid systems. HyDE is being currently used on several projects and has been successfully demonstrated in real-time scenarios. His other research interests are in active diagnosis, diagnosability, measurement selection, and fault adaptive control.

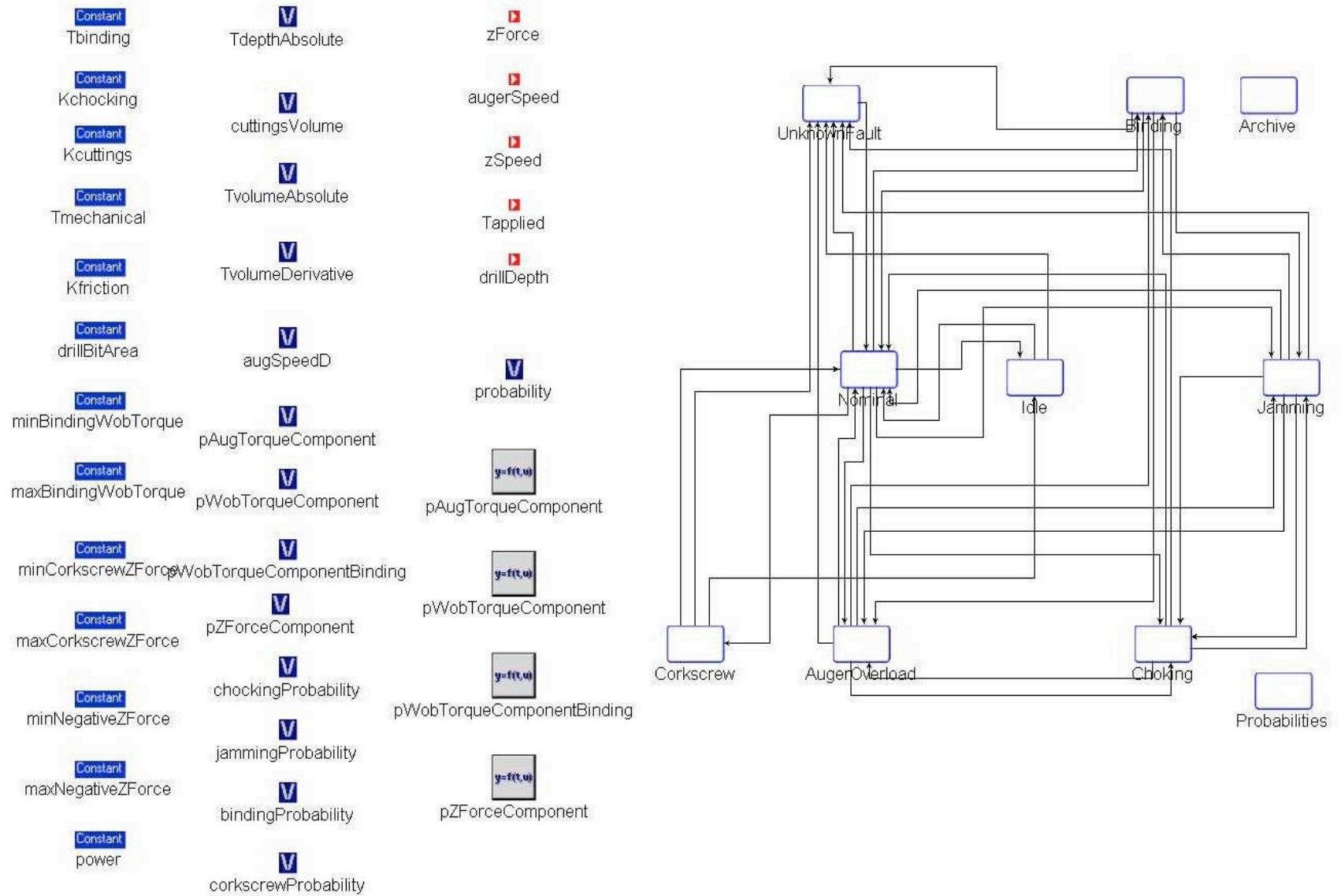


Lee Brownston received his Ph.D. in experimental psychology from the University of Minnesota (1977) and his M.S in computer science from Florida International University (1982). He has worked on production systems at Carnegie Mellon University, blackboard systems at Stanford University, and model-based diagnosis at NASA

Ames Research Center. He has also worked on database applications and web development. He is a co-author, with Robert Farrell, Elaine Kant and Nancy Martin, of "Programming Expert Systems in OPS5: An Introduction to Rule-Based Programming" (1985).

APPENDIX A – DRILL AUGER MODEL IN GME

(high level)



APPENDIX B – DRILL BIT MODEL IN GME

(high level)

13

